

**REMARKS**

Reconsideration and allowance are requested.

The Abstract has been amended as requested.

The Examiner objects to and rejects claims under 35 U.S.C. §112, second paragraph noting several terms in the claims that require clarification. Applicants have amended the claims following the Examiner's suggestions. Withdrawal of the claim objection and the rejection under 35 U.S.C. §112, second paragraph are requested.

The technology described in this application is concerned with handling nested interrupt requests in a data processing system. Of particular concern is managing the priority levels associated with nested interrupt requests. Interrupt priority levels allow later arriving interrupt requests that have a higher priority to "jump the queue" for the processor's attention. Such a "queue jump" then naturally leaves the original interrupt request unfinished. Multiple unfinished interrupt requests (or simply "interrupts") are called "nested" interrupts.

The priority levels associated with different interrupts may be dynamically altered. Such alterations allow the system to respond to changing operating conditions. A high priority interrupt might be dropped to a lower priority as a result of factors either internal or external to the processor. If this lowered priority interrupt is the currently executing interrupt, then nested interrupts below this interrupt in the nest may now have a higher priority, in which case, they are "trapped." A newly-arriving interrupt of higher priority than the recently-lowered, currently-executing interrupt will pre-empt that interrupt and take over as the currently executing interrupt, further delaying the trapped, higher priority interrupt in the nest. Alternatively, an interrupt which was previously low in priority, nested below other interrupts, might have its priority level

increased, yet remained trapped underneath the currently executing interrupt, which is now lower in priority. See page 3, line 20 to page 4, line 2 and Figure 4.

To overcome these problems, the priority of a new interrupt is compared with the priorities throughout a nest of active interrupts and not just with the currently-executing interrupt. Effectively, the nested stack of interrupts inherits as a whole the priority level of its highest priority interrupt. See page 4, lines 4-12.

Turning to the claims, the Examiner's indication that claims 7-10 and 18-20 include allowable subject matter is appreciated. For the reasons explained below, all the claims are allowable.

Claims 1-6 and 11-17 stand rejected under 35 U.S.C. §103 for obviousness based on USP 5,410,715 to Ishimoto and USP 6,081,867 to Cox. This rejection is respectfully traversed.

The Examiner admits that Ishimoto fails to disclose that "a priority of a given active interrupt handling program is alterable whilst said given active interrupt handling program is started and uncompleted." However, Applicants disagree with the Examiner's contention that the remainder of independent claim 1 is disclosed by Ishimoto.

The final two paragraphs (i) and (ii) of claim 1 describe permitting or preventing interrupt pre-emption (i.e., "queue jumping" for the processor's attention) depending on a comparison of the priority level of (1) a pending interrupt program and (2) "a highest priority associated with any of said plurality of active interrupt programs." In other words, a new interrupt request is either dealt with immediately or queued for attention ("nested") depending on whether its associated priority level is higher or lower than the highest priority level of all the currently nested interrupts.

Ishimoto discloses an "interrupt controller with selectable interrupt nesting function." Ishimoto uses indicator flags to designate whether nesting is allowed for a particular interrupt request. Those flags are used to direct whether a new interrupt request must be equal to or higher than the currently executing interrupt in order to preempt it and become the currently executing interrupt itself. As the Examiner admits, pending interrupt priorities are not altered in Ishimoto. In Ishimoto's system, it is only necessary for the priority of a new interrupt request to be compared with the currently executing interrupt because pending interrupt requests nested behind the currently executing interrupt are of lower priority by definition. Consequently, interrupts cannot be "trapped" in the system of Ishimoto.

Hence, Ishimoto does not disclose, and would have absolutely no reason to contemplate, determining whether to permit/prevent a new interrupt request preempting a nested stack of pending interrupts by a comparison of its priority "with the highest priority associated with any of [the nested stack of interrupts]" as recited in paragraphs (i) and (ii) of claim 1. Ishimoto has only one comparison to make between the priority of the new interrupt and the priority of the currently executing interrupt (see column 7, lines 30-52). The sections of text in Ishimoto referred to by the Examiner as describing paragraphs (i) and (ii) of claim 1 only describe nesting in general (column 1, lines 31-40) and the flag which determines whether a new interrupt request must be equal or higher priority in order to preempt the currently executing interrupt (column 7, lines 60-63 and column 10, lines 23-28).

The Examiner cites Cox as teaching a software configurable technique for prioritizing interrupts. This is achieved by the circuit illustrated in figures 3A and 3B, where the interrupt vector address 202-212 which corresponds to the highest priority (as defined by configuration

registers 102-110) interrupt is passed to the CPU. The hardwired gate structure of this circuit ensures that the highest priority interrupt is serviced by the CPU at any given moment.

Cox provides no teaching with regard to nesting interrupt requests. Cox, like Ishimoto, operates on the principle of a constructed hierarchy of interrupt priorities. The circuitry of Figures 3A and 3B ensures that whichever interrupt has the highest priority is the currently executing interrupt. Thus, neither Cox nor Ishimoto ever recognizes the currently-executing interrupt as anything other than the highest priority interrupt.

Accordingly, the proposed combination Ishimoto and Cox, even if it could be made, simply teaches performing a priority comparison for a new interrupt request with the currently executing interrupt. Neither describes making any other comparison with the other nested interrupts. They clearly do not teach permitting a pending interrupt handling program to pre-empt [or permit pre-empting] a plurality of active interrupt handling programs “if a priority associated with said pending interrupt handling program is higher [or less than] than a highest priority associated with any of said plurality of active interrupt programs.” This is not surprising given that Ishimoto and Cox both fail to address the problem with which the claims are directed—let alone the claimed solution to that problem.

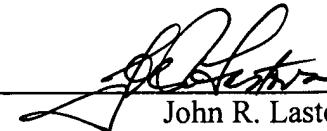
The application is in condition for allowance. An early notice to that effect is requested.

KIMELMAN et al  
Appl. No. 10/775,334  
May 23, 2006

Respectfully submitted,

**NIXON & VANDERHYE P.C.**

By:



---

John R. Lastova  
Reg. No. 33,149

JRL:maa  
901 North Glebe Road, 11th Floor  
Arlington, VA 22203-1808  
Telephone: (703) 816-4000  
Facsimile: (703) 816-4100